Arduino Graphing Barometer for Fishing

Nick Cinquino, Paul Klimah, Carl Klimah 3/30/2016

What environmental factors affect fish activity?
There are as many theories about when the fishing will be good as there are fishermen!
Just a few of the many variables that probably come into play include: light intensity, wind and waves, lunar and solunar phases, time of day, barometric pressure, tides and water temperature (thermocline) at depth.

Several experienced fishermen we know will swear by an association between barometric pressure and fish activity. This holds that fishing is best when barometric pressure is high and stable for a few days. Fishery science has shown that fish can detect barometric pressure (Stoner 2004). Walleye and Crappie sometimes increase their activity level (feeding) and change the depth they occupy during barometric pressure changes (Markham 1991; Guy 1992; Jeffrey and Edds 1999). Around spawning declining barometric pressure has been shown to increase Rainbow Trout and Walleye activity (Peterson 1972; Jeffrey and Edds 1999). Still some studies have found no correlation between barometric pressure and feeding rates (Speers 2012). This is likely due to many possible confounding factors, and interrelationships such as waves to wind relation to pressure, that it's difficult to prove a definite relationship. Clearly though fish react to barometric pressure! It’s up to you to unlock all of sciences secrets and catch some fish while doing it!

We describe an Arduino microcontroller-based barometric pressure measuring circuit that takes a pressure reading each hour, and presents the data graphically as a scrolling chart on a 1.8" color TFT LCD screen (128 x 160).

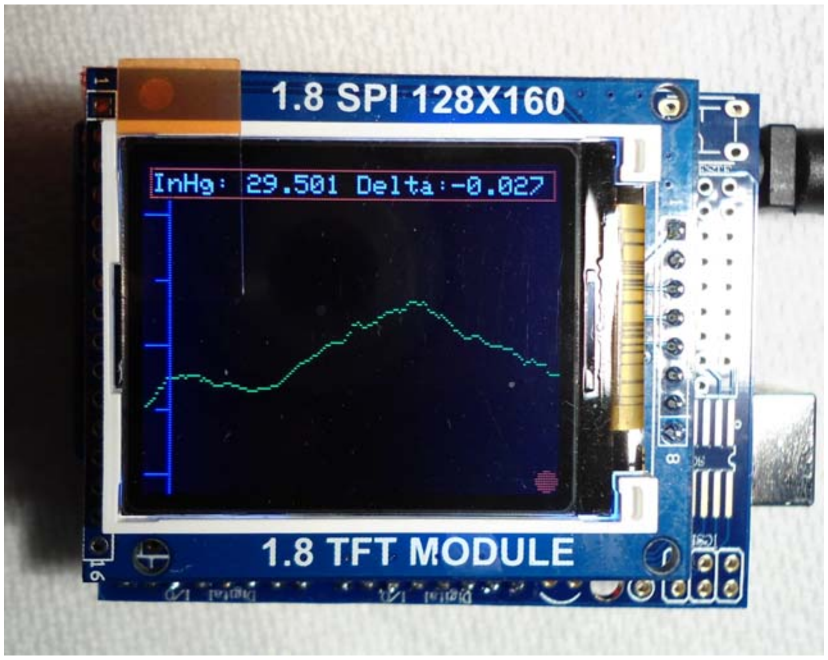
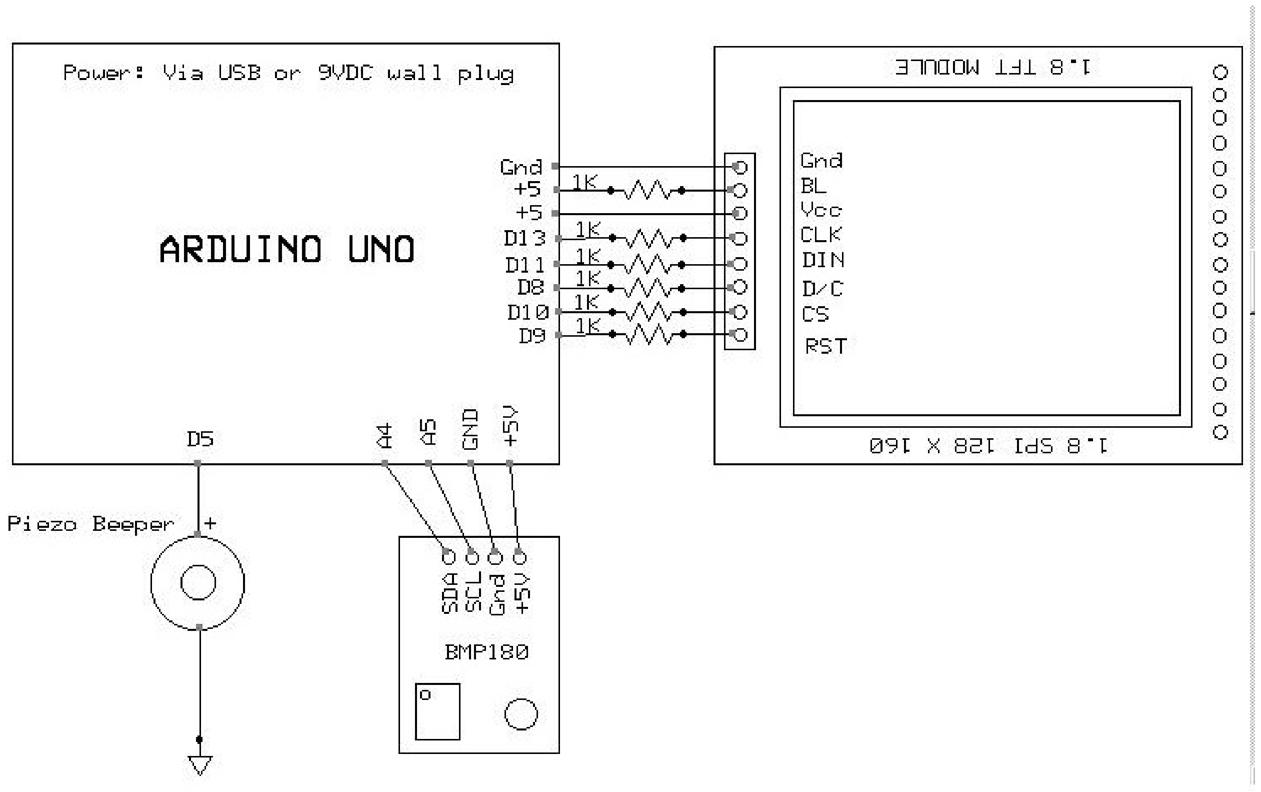


Fig.1: Arduino Fishing Barometer, 6.6 days of data. Far left, pressue dropping fast, red indicator on, thunderstorms arrived within 2 hours on 3/15/16.

Schematic, circuit information and Arduino sketch follows:



Fig, 2: Schematic of the Graphing Fishing Barometer

The circuit is very simple…the real work is being done by the program that controls the display, pressure sensor and piezo beeper. The pressure sensor is a typical BMP180 digital module with Bosch sensor, available from multiple sources such as Adafruit, Sparkfun, Aliexpress or Banggood. The piezo beeper can be any type but keep it small to fit underneath the LCD module, see photos. The LCD module should be the 1.8" TFT LCD, 128 X 160 pixels, and the driver chip should be the ST7735. When searching for it, try to match the unit to the one shown in the photos and diagrams. The piezo beeper is a small 5V unit, which delivers an hourly audible indication of barometric status as well.

Construction is easy as well; acquire a "screw shield board" and solder a row of 8 female stacking headers to the shield board as shown in the photo, for connection to the LCD module. Solder pin headers to the board for connection to the Arduino. Add the resistors, and add soldered jumpers from the module to the board per the schematic. Add the piezo beeper in a location that does not interfere with anything on the bottom of the LCD module. In the photo, it is also used to support the display board; prevent it from being bent down. Solder the BMP180 module and add jumpers per the schematic. If the beeper is unwanted, replace with a 470Ohm resistor and an LED.

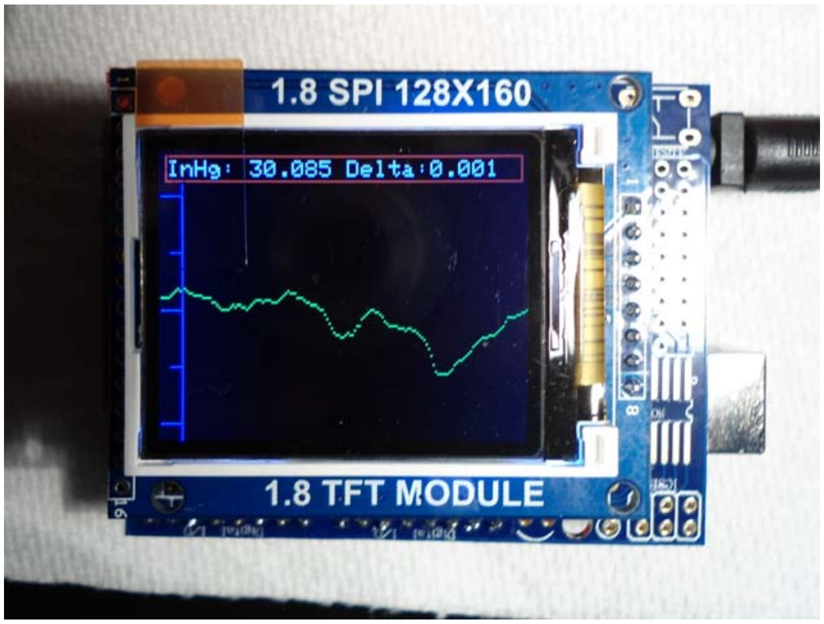


Fig.3: Another in-use screenshot. 6.6 days of data displayed. The dip at right was rain.

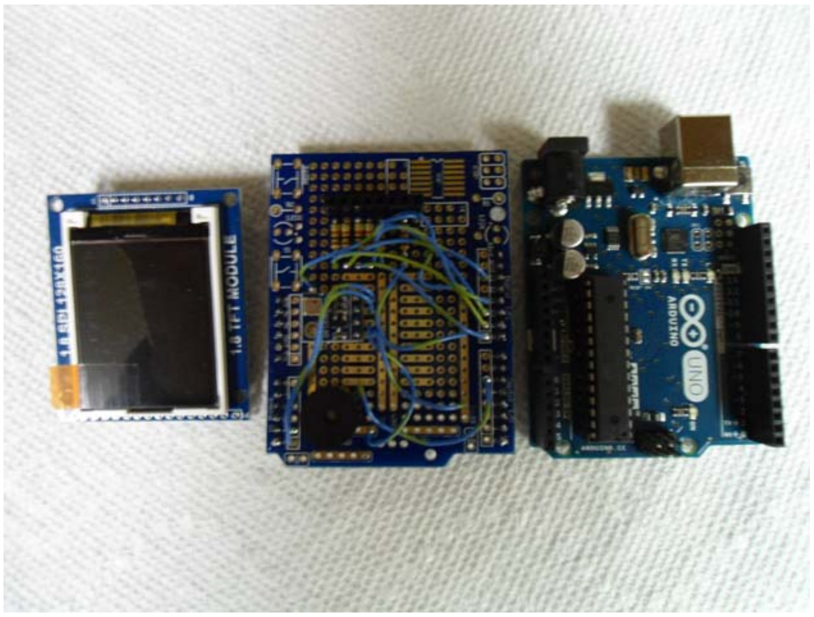


Fig.4: The 3 layers pulled apart, LCD module, screw shield with header, BMP180 and beeper, and Arduino Uno

Arduino sketch: Start by adding the required library, Adafruit_BMP085.h to your library file from Github.com (Ref.12). Then copy and paste the sketch into your Arduino program window. If it verifies, assemble the barometer and load the sketch. Open the serial window to verify BM180 function.
IMPORTANT: The first code line under void loop() is "myAltitude"…830 is the altitude in feet above sea level of one author's location…delete 830 and enter YOUR altitude above sea level in feet. Use your nearest airport altitude or look up various cities in (Ref.11).

```
/*
```

```
 Fishing Barometer NJC 1/26/16, 1.8" TFTLCD SPI display.
 RST=9, CS=10, D/C=8, DIN=11, CLK=13 (1Kohm each), Vcc, BL +5, Gnd=Gnd
 Piezo beeper=D5, BMP180:SDA=A4, SCL=A5, Vcc=+5, Gnd=Gnd
 Sampling every 60 minutes.
 This program draws a scrolling Barograph on a TFT screen.
 Pressure is shown as Inches of Mercury, adjusted to Sea Level.
 The graph propagates (scrolls) from
 left to right as the sensor reading is updated.
 The sketch was written and tested on a TFT display sized 160x128 pixels with ST7735.
 */

#include <Wire.h>   //wire library
#include <Adafruit_BMP085.h>  //pressure sensor library
#include <TFT.h>  // Arduino LCD library
#include <SPI.h>  //serial peripheral library
#define cs   10   // pin definitions from LCD to Uno
#define dc   8
#define rst  9
Adafruit_BMP085 bmp;
TFT TFTscreen = TFT(cs, dc, rst);
int wTFT = TFTscreen.width();  //define TFTscreen.width() as wTFT
int hTFT = TFTscreen.height(); //define TFTscreen.height() as hTFT
int count;
int sensorArray[161]; //array setting
float inhg;
int scaleval;
float oldinhg;
float delta;
float loval;
float variance = 0.025;  //adjust rate of pressure change value for 1 hour
int beep = 5;  //beeper on pin5
int timer;
float myAltitude;
float SLPfactor;


void setup(){
  Serial.begin(9600);   // initialize the serial port
  Serial.println("Pa Station,  InHg,  Y-Value,  Delta,  Millis");
  pinMode (beep, OUTPUT);   //pin 5 = beeper
  TFTscreen.begin();  // initialize the display
  TFTscreen.background(0, 0, 0);  // set TFT background color to black (erase)
  TFTscreen.stroke(0,255,0);  // set the font color to green
  TFTscreen.setTextSize(3);  // set the font size
  TFTscreen.text("Gone",30,40);   // write the text to the screen
  TFTscreen.stroke(0,255,0);  // set the font color to green
  TFTscreen.setTextSize(3);  // set the font size
  TFTscreen.text("fishin'!",10,70);   // write the text to the screen
  delay(2000);
  TFTscreen.background(0, 0, 0);  // set TFT background color to black (erase)
  for (int count=0; count<=159; count++) //initialize the array & counts up 0 to 159
  {
    sensorArray[count] = 0; //this 'zeros' all array elements(?)
  }
  if (!bmp.begin()) {
    Serial.println("Could not find a valid BMP085 sensor, check wiring!");
    while (1) {
    }
  }
}


void loop() {
  myAltitude = 830;  //ENTER YOUR ALTITUDE IN FEET ABOVE SEA LEVEL, to nearest foot.
  SLPfactor=myAltitude/996;  //Sea level factor assumes linear 996' per InHg
  inhg = (bmp.readPressure()) / 3386.3; //read pascals, convert to InHg
  inhg = inhg + SLPfactor ;  //convert pascals to InHg, then add Sealevel correction
  delta = inhg - oldinhg;  //find hourly change in pressure
  loval = oldinhg - inhg;  //find low pressure change value
  scaleval=(inhg * -50) + 1570;  //convert SLPInHg to y-axis of LCD to match gridlines
  sensorArray[0] = scaleval; //this stores current drawHeight() value on the 1st element of array
  Serial.print((bmp.readPressure()));  //serial window data printouts
  Serial.print(",  ");
  Serial.print(inhg, 3);
  Serial.print(",  ");
  Serial.print(scaleval);
  Serial.print(",  ");
  Serial.print(delta, 3);
  Serial.print(",  ");
  Serial.println(millis());


  for ( int count = 1; count <= wTFT; count++ )  //counts up from 1 to 160 (wTFT)
  {
    TFTscreen.stroke( 0, 0, 0 ); //set color to black (erase)
    TFTscreen.line(count, 0, count, hTFT );  //simulate erase by making entire column black

    TFTscreen.stroke( 0, 255, 0 ); //set color to green pixel
    TFTscreen.point(count, sensorArray[count-1] );  //draw green color pixel
  }  //end of dot array print 'for' loop


  if (delta >= variance ) {
    TFTscreen.fillCircle(155, 123,4,ST7735_RED);  //color messed up in library-high P
    digitalWrite(beep, HIGH);  // beep
    delay(160);
    digitalWrite(beep,LOW);
    delay(160);
    digitalWrite(beep, HIGH);  //double beep
    delay(160);
    digitalWrite(beep,LOW);
  }
  else if (loval >= variance ) {
    TFTscreen.fillCircle(155, 123,4,ST7735_BLUE);  //low P
    digitalWrite(beep, HIGH);  // beep
    delay(96);
    digitalWrite(beep,LOW);
    delay(96);
    digitalWrite(beep, HIGH);  //double beep
    delay(96);
    digitalWrite(beep,LOW);
    delay(96);
    digitalWrite(beep, HIGH);  //double beep
    delay(96);
    digitalWrite(beep,LOW);
  }
  else {
    TFTscreen.fillCircle(155, 123,4,ST7735_BLACK); //no change
    digitalWrite(beep, HIGH);  // beep
    delay(480);
    digitalWrite(beep,LOW);
  }

  TFTscreen.stroke(255,255,255);  // set the font color to white
  TFTscreen.setTextSize(1);  // set the font size
  TFTscreen.setCursor(4,4);
  TFTscreen.text("InHg: ",4,5);   // write the text to the top left corner of the screen
  TFTscreen.drawRect(3,3,155,12,ST7735_BLUE);  //draw red rectangle for numerical data
  TFTscreen.print(inhg,3);  //numerical data to screen
  TFTscreen.print(" ");
  TFTscreen.print("Delta:");
  TFTscreen.print(delta,3);

  TFTscreen.drawLine(10,15,10,127,ST7735_RED);  //gridlines, rem:BLUE
  TFTscreen.drawLine(10,20,1,20,ST7735_RED);
  TFTscreen.drawLine(10,45,5,45,ST7735_RED);
  TFTscreen.drawLine(10,70,1,70,ST7735_RED);
  TFTscreen.drawLine(10,95,5,95,ST7735_RED);
  TFTscreen.drawLine(10,120,1,120,ST7735_RED);


  for (count=wTFT; count>=2; count--) // count down from 160 to 2
  {
    sensorArray[count-1] = sensorArray[count-2]; //count=160, count=159 ,count=3 , etc
  }

  for(timer = 0; timer < 3597; timer += 1)  //time delay loop, 3597=1hr, adjust as needed.
  {
    delay(1000);  //1sec loop delay
  }

  if (timer >=3597){   //reset time delay loop, 3597=1hr, adjust as needed.
    timer = 0;
  }
  oldinhg = inhg;  //save previous pressure value to compare to present
}
```

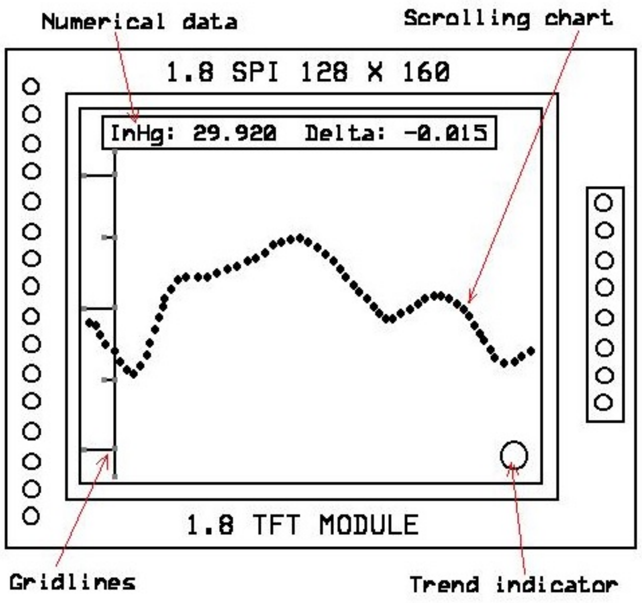


Fig.5: Data layout on the 1.8", 128 X 160 TFT LCD display module.

At the top of the display, inside a red rectangle, is the current numerical pressure, from the present hour, relative to Sea Level in Inches of Mercury, and the numerical change in pressure over the last hour. The blue gridlines at far left are calibrated for 29.0, 30.0 and 31.0 InHg for the 3 longer horizontal lines. These values were chosen since in USA the Sea Level pressure "rarely exceeds 31 InHg, and rarely falls below 29 InHg" (Ref.9). There is a little extra headroom at both top and bottom. A hurricane would therefore be offscale at the bottom; if there is a hurricane in the area, one ought not to be fishing! The Standard Pressure of 29.92 InHg is about center-scale. In the center is the green scrolling display, from left to right, updated hourly. At bottom right is the trend indicator: red if pressure is falling faster than 0.025 InHg/hr or blue if pressure is rising faster than 0.025 InHg/hr. The piezo beeps once, hourly, when pressure is stable (change is <+/- 0.025 InHg/hour). 2 quick beeps indicates pressure is rising faster than +0.025 InHg/hour. 3 quick beeps indicates pressure falling faster than -0.025 InHg/hour. So, even if you're not looking at the unit frequently, it communicates an audible barometric update as long as you're close enough to hear the beeps.

Another helpful tool to help formulate a longer range prediction of local barometric pressure trend and therefore fish activity is the North American Surface Analysis Chart from NOAA (Ref.10). Weather systems generally move west to east, so if you note a significant low pressure west of your location, watch for the beginnings of a drop in pressure in the coming days on the barometer display. Besides its usefulness in trying to predict fish activity, this project may also come in handy for identifying the cause of sudden migraine or sinus headaches, joint discomfort, and for general meteorological study.

In use, about 1 week before a planned fishing trip, power up the unit and keep an eye on the developing pressure changes , especially in the 2-3 days immediately prior to the trip. When first powered on, all array values are zero so there will be a flat line along the top that will be replaced with barometric values with time. Full scrolling begins after 6.6 days.

REFERENCES

Speers 2012- http://mspace.lib.umanitoba.ca/handle/1993/260

Stoner 2004- http://onlinelibrary.wiley.com/doi/10.1111/j.0022-1112.2004.00593.x/full

Markham 1991- http://www.tandfonline.com/doi/abs/10.1577/1548-8675(1991)011%3C0504%3AWCSMAH%3E2.3.CO%3B2

Guy 1992- http://www.tandfonline.com/doi/abs/10.1080/02705060.1992.9664679

Jeffrey and Edds 1999-
http://www.tandfonline.com/doi/abs/10.1080/02705060.1999.9663694#.Vs92V_krKUk

Peterson 1972- http://www.tandfonline.com/doi/abs/10.1577/1548-8640%281972%2934%5B110%3ABPAIEO%5D2.0.CO%3B2?journalCode=uzpf20

7) https://weather.com/sports-recreation/fishing/news/fishing-barometer-20120328

8) http://www.in-fisherman.com/bass/barometric-pressure-and-bass/

9) http://www.infoplease.com/cig/weather/pressure-getting-to-you.html

10) NOAA surface analysis: http://www.wpc.ncep.noaa.gov/html/sfc2.shtml

11 http://egsc.usgs.gov/isb//pubs/booklets/elvadist/elvadist.html

12) https://github.com/adafruit/Adafruit-BMP085-Library